Application No.: 09/911819 Docket No.: MWS-077

REMARKS

Claims 1-56 were presented for examination. No claims have been added or amended. No new matter has been added. Claims 1-56 are now pending. Claims 1, 11, 21, 29, 39, and 49, are independent.

Rejections Pursuant to Judicially Created Doctrine of Obviousness-type Double Patenting

The Examiner provisionally rejected claims 1, 11, 29, and 39, under the judicially created doctrine of obviousness-type double patenting as unpatentable over claim 16 of copending Application No. 10/190288 (hereafter '288). Applicants respectfully traverse the rejection for double patenting for the reasons set forth below.

The Examiner stated that the claims are not patentably distinct from each other. Claim 16 of '288 depends on claims 15, 14, 13, 12, 11, 8, 3, and 1, also of '288. Therefore, claim 16, with all of its limitations from the claims on which it depends, recites:

An automated method of generating a software component for use by an application program comprising:
receiving source code for one or more functions created in a first programming environment;
processing the source code to create a component including object code for the one or more functions, (from Claim 3)
wherein the component comprises a Component Object Model (COM) object (Claim 3),
wherein processing comprises:
converting the source code from a first programming language to a second programming language; (from Claim 8)
generating COM source code files for the component in the second programming language (from Claim 8);
wherein the COM source code files include an Interface Description Language (IDL) source code file, class definition and implementation files, a DLL exports files, and a source file providing an implementation of each exported function of the component (from Claim 11); and
wherein the generated COM source code files further include a resource script file [and] wherein the IDL compile further produces a component type library file (from Claim 16);
invok[ing] an IDL compiler to process the source code file to produce an interface header file and an interface GUID file (from Claim 12);
compiling the converted source code files, generated COM source code files and processed IDL files to produce object files (from Claim 13);

wherein compiling compiles a file that includes template implementations of all required COM base classes and registration code (from Claim 14),

linking one or more libraries required to support the one or more functions to the object files to produce a version of the component that does not include type information (from Claim 15);

invoking a resource compiler on the version of the component that does not include type information, the resource script file and the component type library file to produce the component (from Claim 16); and

wherein the component is usable by the application program in a second programming environment to access the one or more functions of the component.

The Examiner admits that the independent claims and claim 16 from '288 are not identical. Applicants respectfully suggest that not only are the claims not identical, they are also patentably distinct. Applicants' independent claim 1 and 29 do not recite any of the limitations on processing recited by claim 16. Neither of Applicants' independent claims recites converting the source code from a first programming language into a second programming language, generating source code files for the component in the second programming language, invoking an IDL compiler to process the source code file, compiling converted source code files, linking libraries, or invoking a resource compiler. Even assuming for the sake of argument that the cited claim 16 in the '288 patent suggests the processing of a definition of a function associated with a first language, the limitations of dependent claim 16 render it patently distinct from independent claims 1 and 29.

Similarly, neither Applicants' independent claim 11 nor independent claim 39 recite the limitations recited by claim 16. Neither independent claim 11 nor independent claim 39 recite any steps directed to processing or compiling, focusing instead on providing a file of description items. Since dependent claim 16 includes limitations relating to processing source code, invoking compilers, converting source code, and compiling converted source code, independent claims 11 and 16 are also patentably distinct from dependent claim 16.

Given that, as set forth above, the current limitations of '288 do not describe the same matter as claims 1, 11, 29 and 39, and given that the claim 16 from '288 may change over the course of its prosecution, at this time, Applicants respectfully traverse the provisional rejection for double patenting.

Summary of Shannon

Shannon et al., "Mapping the Interface Description Language Type Model in C," November 1989, IEEE Transactions on Software Engineering, Vol. 15, No. 11 (Shannon), discusses an interface description language (IDL) standard, Diana. (See Shannon, at 1333). Diana is an intermediate representation of Ada programs, used within Ada compilers. (See Shannon, at 1333). Diana functions as an extension of the Ada language, extending Ada structures. (See Current Office Action, Examiner's Note, page 5). A translator tool maps descriptions representative of Ada into code fragments written in the target programming language. (See Shannon, at 1333). Shannon focuses on using an IDL, like Diana, to map Diana structures to the C programming language. (See Shannon, at 1333).

Summary of Research Systems

Research Systems, "IDL," copyright 1994 (Research Systems), provides a language in which a scientist or scientific application developer may prototype and develop an application. (See Research Systems, page 1). Research Systems' IDL combines mathematics, advanced data visualization, scientific graphics, and a graphical user interface toolkit to analyze and visualize scientific data. (See Research Systems, page 1). The IDL is a complete structured language and provides or supports math functions, 2D plotting, surface plotting, 3D graphics, mapping functions, image processing, general graphics, graphic output devices, GUI toolkits, signal processing, statistics, and various input/output formats. (See Research Systems, pages 1-5).

Summary of Elmroth

Elmroth et al., "A Web Computing Environment for the SLICOT Library," December 2000, Brite-Euram III, Networks Programme NICONET (Elmroth), presents a prototype web computing environment for computations related to the design and analysis of control systems using the Subroutine Library In Systems and Control Theory (SLICOT) software library. (See Elmroth, Abstract and page 8). In this environment, a user supplies input data to the SLICOT routine that will be executed. (See Elmroth, Abstract and page 7). The routine then computes output data, the user receives the output data, and the data is stored on a server. (See Elmroth, Abstract and page 7). The environment enables the testing and use of SLICOT routines, which

may be executed on a remote web server with no need for software installation on a local computer, allowing a user to test the routines with little effort. (See Elmroth, Abstract and page 8).

Rejections Pursuant to 35 U.S.C. §103(a)

The Examiner rejected claims 1 and 29 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest processing a function written in a source code language to create description information about that function.

Claims 1 and 29 include the step of processing a function written in a source language and that processing results in description information, which a user may later translate into any number of target languages. (See claims 1 and 29). Shannon teaches a method for writing a program in terms of a target language. Shannon does not disclose writing a program in the target language, but neither does Shannon disclose processing a function written in a source language to create description information usable in translating the source language into a target language. The Examiner admits that Shannon does not disclose processing a definition of a function associated with a source language to create description information, but the Examiner states that Shannon suggests this processing by disclosing a method comprising creating description information about a function in a first language using the interface description language (IDL) for the description information. (See Current Office Action, page 5). However, in Shannon, the disclosed IDL notations are data declarations and utilities phrased in terms of the target language. (See Shannon, at 1333). A user of the system disclosed by Shannon writes a program using IDL and the IDL translator then maps the structures in the program into the target language. (See Shannon, at 1333 and 1336). Since the user of the Shannon system does not write in a source language, but must instead write programs using the data declarations and utilities, which are representative of the target language, Shannon does not teach or suggest processing a function written in a source code language to create description information about that function.

The Examiner also states that if Shannon does not teach processing, it would have been obvious to use the teaching of Research Systems to do so. (See Current Office Action, page 6). However, Research Systems fails to teach or suggest processing a function written in a source

language to create description information, which is neither in the source language nor in a target language. A user of the Research Systems programming language writes in that language and uses that language to develop an application in a target language. Research Systems does not suggest processing a program, or even a function, written in a source language to generate description information, which a user may translate into a target language.

The Examiner rejected Applicants' claims 2, 4, 24-25, 30, and 32 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest creating, storing, or using description information when processing a function in a source language.

Claims 2, 4, 24-25, 30, and 32 include the elements of creating, storing, or using description information created during the processing done in the method of claim 1. The Examiner states that Shannon discloses a file of description items and derived description information. (See Current Office Action, page 6). However, since Shannon discusses items and information originally written in IDL and not derived from processing code written in a first source language, Shannon does not disclose the claimed file of Applicants' invention.

The Examiner rejected Applicants' claims 3, 23-25, 31, and 51-53, based on Shannon and Research Systems. Applicants respectfully traverse this rejection. Shannon and Research Systems fail to teach or suggest the creation of description information about a function in a source programming language.

Claims 3, 23-25, 31, and 51-53 include the elements of using derived information about a function, and of translating a call to the function into a call to a corresponding function in a second language. (See claims 3, 23-25, 51-53). The Examiner states that Shannon and Research Systems combine to disclose using derived information about the function to translate the call to the function in a call to a corresponding function. (See Current Office Action, page 6). However, Shannon and Research Systems discuss translating a function call from a first language (IDL Specification) to a second language (C). Writing a program in one language and translating the program into a second language does not suggest taking a function already written in one language, creating description information about the program, and then translating the description into a target language. Examining how a graphical or mathematical function is

defined does not suggest the creation of description information about a function in a source programming language.

The Examiner rejected Applicants' claims 5 and 33 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest storing a translated function into a library of entries.

Claims 5 and 33 include the elements of storing a translated function in a second language in a library of entries. The Examiner states that Shannon discloses creation of C language constructs, implicitly disclosing the .lib files associated with the assembling of object files prior to linking in C. (See Current Office Action, page 7). However, simple creation of a language construct, even if the construct is a library, does not teach storing a translated function into a library of entries.

The Examiner rejected Applicants' claims 6-7, 12-13, 20, 34-35, 40-41, and 48 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest translating of parameter in a source language into description information, where the description information is used in a later translation into a target language.

Claims 6-7, 12-13, 20, 34-35, 40-41, and 48 include the elements of deriving a number of declared form inputs and outputs, or including a descriptor identifying known input and output arguments, as part of processing a function. The Examiner admits that Shannon does not disclose processing of the function and deriving a number of declared formal inputs and outputs to the functions. (See Current Office Action, page 7). The Examiner takes Official notice that the declaration of formal input and output, and scope of variables declared in a function in 4th generation like C was a known concept at the time the invention was made; hence the IDL input/output provision suggested by Shannon implies analysis of the first language input/output formal parameters leading to creation of C formal parameters. (See Current Office Action, page 7). However, analyzing a source language input/output parameter for creation of input/output parameters in a target language fails to teach or suggest the translating of parameter in a source language into description information, where translating the parameter does not result in creation of another parameter in a target language but into a description used in a later translation into a target language.

The Examiner rejected Applicants' claims 8, 14-16, and 36, 42-44 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest description of a function scope for use in later translation into a target language.

Claims 8 and 36 include the step of deriving a scope of the processed function. Claims 14 and 42 include the element of a descriptor identifying a scope of the processed function. Claims 15-16 and 42-44 include the element of a descriptor identifying an acceptance of a variable input argument list into a function or a return of a variable output argument list from a function. The Examiner states that namespace and function scope involving local and global parameters are known in 4th generation like C as suggested from the above Official notice. (See Current Office Action, page 7). However, the mere existence of namespace and function scopes does not render it obvious to analyze source language to derive a scope, or variable input and output argument lists, for use in description information describing the source language, where the description information does not involve the creation of a scope but extraction of the scope and description of the scope for use in later translation into a target language.

The Examiner rejected Applicants' claims 9-10, 14-16, 18-19, 37-38, and 46-47 based on Research Systems. Applicants respectfully traverse this rejection. Research Systems fails to teach or suggest processing of a definition of a function to create description information.

Claims 9-10, 37-38 include the step of determining whether a processed function accepts a variable number of arguments or returns a variable number of results. Claim 14 includes the element of a descriptor identifying a scope of the processed function. Claims 15-16 include the element of a descriptor identifying an acceptance of a variable input argument list into a function or a return of a variable output argument list from a function. Claim 18-19 and 46-47 include the steps of generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function or a return of a variable output argument list from the function.

The Examiner admits that Shannon, even in view of Research Systems, does not teach determining variable arguments in a function and a variable return of results. (See Current Office Action, page 7-8). The Examiner takes Official notice that the advanced languages like C providing a variable number of arguments was known at the time the invention was made. (See

Current Office Action, page 7-8). However, as noted above, the combination of references does not teach or suggest the processing of a definition of a function to create description information. Therefore, the combination of references must fail to teach or suggest the steps of processing included in these claims.

The Examiner rejected Applicants' claims 11, 24-25, and 39 based on Shannon. Applicants respectfully traverse this rejection. Shannon fails to teach or suggest processing a function in a source language to create description information, used later for translation of the function into a target language.

Claims 11 and 39 are methods including the steps of providing a file of description items and using the file of description items to translate a first program file into a second program file. Claim 24 includes the step of using derived information about each function to create description information and including that information in a description file. Claim 25 includes the steps of retrieving description information about the function from the description file, and using the description information to translate the call to the function in the first language into a call to a corresponding function in the second language. Shannon discusses using a file to translate into the target language without requiring processing of the definition of functions in the file and using the file description items to translate a function a second program file. The Examiner states that Shannon therefore discloses providing a file of description items, information about a function in a first language, the description being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language. (See Current Office Action, page 8). However, as noted above, Shannon discusses providing description information in a first language, but the claimed invention processes a function in a source language to create description information, used later for translation of the function into a target language.

The Examiner admits that Shannon fails to disclose using information about a function associated with the first language to translate a first program file into a second program file but states that in view of Shannon and Research Systems, this use of information about a function would have been obvious. (See Current Office Action, page 8). The combined teachings of Shannon and Research Systems fail to disclose the integral step of processing a source language

to create description information and using the description information to translate the source language into a target language.

The Examiner rejected Applicants' claims 17 and 45 based on combining Shannon and Research Systems. Applicants respectfully traverse this rejection. Shannon and Research Systems fail to teach or suggest translating a function from a source language into description information for translating the description information into a target language.

Claims 17 and 45 include the steps of retrieving an item from a file of description items for each call to a function in the first program file, using the description information in the item to translate the call to the function in the first language into a call to a corresponding function in the second language, and storing the translated function in the second program file. The Examiner admits that Shannon does not disclose for each call in the first program file retrieving an item from the file of description items and using information description in the item to translate the first language function into a call corresponding to the second language; and storing the translated function in the second program file. (See Current Office Action, page 9). The Examiner states that in view of teachings by the combination of Shannon and Research Systems, the above limitation is implicitly disclosed. (See Current Office Action, page 9). However, as discussed with regard to claims 1 and 11, it is not implicit from combining Shannon and Research Systems to translate a function from a source language into description information for later translating the description information into a target language. Neither is it implicit to complete such a translation through retrieval of an item from a file of description items.

The Examiner rejected Applicants' claims 21, 22, and 49-50 based on Elmroth, in view of Research Systems and further in view of Shannon. Applicants respectfully traverse this rejection. Elmroth, Shannon, and Research Systems fail to teach or suggest processing a function written in a source language to create description information and then translating the description information into a target language.

The Examiner states that Elmroth discloses a method comprising providing a library file including functions defined by a first language, processing the library file to create a function library and a description file, and using the description file to translate a program file from the first language into the second language. (See Current Office Action, page 10). The Examiner

admits that Elmroth does not disclose the description file including description information being sufficient to enable translation of a call to the function into a call to a corresponding function in a second language without processing of the definition of the function. (See Current Office Action, page 10-11). The Examiner suggests that it would have been obvious to use Elmroth for this purpose in light of Shannon and Research Systems. (See Current Office Action, page 11). However, as discussed above, neither Shannon nor Research Systems teach or suggest processing a function written in a source language to create description information and then translating the description information into a target language. Since the Examiner admits that Elmroth would not teach or suggest such translation either, combining Elmroth with Shannon and Research Systems fails to teach or suggest the claimed invention.

The Examiner rejected Applicants' claims 26-28, and 54-56 based on Elmroth, in view of Research Systems and further in view of Shannon. Applicants respectfully traverse this rejection. Elmroth, Shannon, and Research Systems fail to teach or suggest generating a call based upon description information derived from a definition of a function.

Claims 26-28 and 54-56 include the steps of generating a call either through a function evaluation interface or through a normal interface based on description information about known and variable input and out argument lists. The Examiner states that in light of the rationale to combine Shannon with Elmroth in claim 21, there exists motivation to combine the Web interface from Elmroth with the function evaluation interface suggested by Shannon. (See Current Office Action, pages 12-13). As discussed for claim 21, even a combination of Elmroth with Shannon and with Research Systems fails to teach or suggest the claimed invention.

Accordingly, since Shannon, Research Systems, and Elmroth fail to teach or suggest each and every element of claims 1-56, Applicants request the withdrawal of the rejections and the allowance of the claims.

Application No.: 09/911819 Docket No.: MWS-077

CONCLUSION

In view of the above amendment, applicant believes the pending application is in condition for allowance.

Applicant believes no fee is due with this statement. However, if a fee is due, please charge our Deposit Account No. 12-0080, under Order No. MWS-077 from which the undersigned is authorized to draw.

Dated: November 3, 2004

Respectfully submitted,

By _____

Kevin J. Canning
Registration No.: 35,470
LAHIVE & COCKFIELD, LLP
28 State Street
Boston, Massachusetts 02109
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Applicant